Algorithms 2024/25: Tick 3

Your task is to implement a solver for a set of ordering constraints on a set of tasks. Implement a Java interface containing one method, `void solve(String filename)`, which must perform the following steps:

1. Open the file with the specified filename. Throw a (java.io.)`IOException` if you cannot open or read from the file.
2. The first line of the file contains the number, n, of tasks to be scheduled, which are referred to by numbers 1..n.
3. All the subsequent lines are of the form `a,b` where a and b are integers in the range 1..n. The pair indicates that task 'a' must be completed before task 'b'.
4. Your program must compute any order of completing the tasks 1..n that does not violate any of the ordering constraints.
    a. If no such ordering exists, your method must print "IMPOSSIBLE" and return.
    b. If a valid ordering exists, your method must print the numbers 1..n to standard output (hint: use `System.out.println`), one per line, in an order (first to last) that your program has calculated to be a valid order. If many valid orderings exist, it does not matter which one your program chooses to print.
5. If your program found an order that does not violate the constraints, it should also print one further line reading "THE LONGEST DEPENDENCY CHAIN IS X TASKS", where X is the number of tasks (not dependencies between tasks) in the longest chain that is constrained by ordering constraints. The value of X must correspond to the optimal schedule given the ordering constraints, and might be described as the 'critical path' when unconstrained tasks could be performed in parallel.

Your implementation should not require any libraries or external code other than to read in the data from the input file (I recommend using utility classes in the java.io and/or java.util packages, and the `Integer.parseInt` static function).

What to submit:

1. Your Java implementation: a single Java source containing a class that implements the interface provided. Helper functions are allowed and encouraged to make your code readable! You may write unit tests and a class containing a `main(String[])` function in other files but are not asked to submit those.

Submit >> here << You may re-submit if you wish. **The deadline is 12:00 on Wed 19 Mar 2025**.

```
interface Algs202425Tick3 {
  void solve(String filename) throws java.io.IOException;
}
```

Save this ^^ into Algs202425Tick3.java, then write your solution in Solver.java:

```
public class Solver implements Algs202425Tick3 {...}
```

Dr John Fawcett, Mar 2025